arXiv:2303.03986v1 [cs.LG] 5 Mar 2023

复用梯度下降：无需反向传播，在硬件神经网络上快速在线训练现代数据集

A. N. McCaughan[1], B. G. Oripov[2], N. Ganesh[1], S. W. Nam[1],A. Dienstfrey[1], S. M. Buckley[1]

[1]美国国家标准与技术研究院，博尔德，科罗拉多州 80305[2]科罗拉多大学博尔德分校，博尔德，科罗拉多州 80309

摘要

我们提出了复用梯度下降（MGD），这是一种梯度下降框架，旨在轻松地在硬件上训练模拟或数字神经网络。MGD利用零阶优化技术在线训练硬件神经网络。我们展示了该方法在现代机器学习数据集（包括 CIFAR-10 和 Fashion-MNIST）上训练神经网络的能力，并将其性能与反向传播进行比较。基于现实的时间尺度和硬件参数，我们的结果表明，这些优化技术能够在新兴硬件平台上训练网络，其速度比通过标准GPU上的反向传播训练所需的实际运行时间快数个数量级，即使硬件存在不完美的权重更新或器件间差异也是如此。此外，我们还说明了如何将其作为芯片在环训练的一部分应用于现有硬件，或直接集成到硬件层级。至关重要的是，MGD框架具有很高的灵活性，其梯度下降过程可以进行优化，以补偿特定的硬件限制，例如较慢的参数更新速度或有限的输入带宽。

1 引言

机器学习已被证明是各种应用中不可或缺的工具 [1]。然而，在传统数字硬件上进行机器学习效率低下，因此人们投入大量精力开发定制硬件，以便以更高速度、更低能量代价执行机器学习任务 [2]。目前已出现多种采用模拟 [3],数字 [4, 5], 或混合信号处理 [6] 的硬件平台，有望提高运行速度和/或降低能量代价 [7]。然而，许多最有前景的硬件实现只能执行机器学习算法中的推理部分。与此同时，大部分能量代价都花在数据集上的训练 [8], ，通常采用梯度下降。反向传播是计算梯度下降所需梯度时最常用的方法，但事实证明，要在新型硬件平台上实现这一方法颇具挑战 [9]。

尽管人们常常将二者混为一谈，但通过梯度下降进行训练并不需要反向传播——反向传播仅用于计算梯度。神经网络中还存在其他计算梯度的方法，但与反向传播相比，这些方法在软件中的效率要低得多，因此在当今的机器学习应用中很少使用。但在硬件领域，情况通常并非如此：反向传播不仅可能难以实现，也可能不是计算梯度的最高效方法。

硬件领域中特别值得关注的是无模型方法。这类方法不需要了解神经网络的内部结构（例如拓扑结构、激活函数、导数等），只需能够扰动神经网络的参数并测量其响应即可。此类方法最简单的例子是有限差分 [10], ，该方法已用于芯片在环训练 [11]。然而，有限差分还有其他若干缺点，阻碍了它在硬件中的广泛应用，其中包括每个突触都需要额外内存，以及需要全局同步。幸运的是，还有多种其他无模型方法可以克服有限差分 [12, 13] 带来的一些问题。

本文表明，无模型扰动方法可用于高效训练现代神经网络架构，而且能够以原生方式在新兴硬件中实现。自20世纪90年代起，人们便开始研究将这些方法用于训练VLSI神经网络 [14, 15, 16, 17, 18, 19, 20, 21, 22, 23], ；近来，这些方法也被应用于忆阻器交叉阵列 [24] 和光子硬件 [25], ，但所有这些演示的规模都非常有限，仅涉及小型数据集和少数神经元。下文将介绍一个框架，说明如何将这些技术应用于规模大得多的现有硬件，并着重构建简单、高度局部化的电路，以便在需要时实现片上部署。该框架还可扩展至通过芯片在环技术训练现有硬件系统。我们指出，这些方法也已结合自动微分应用于前向梯度方法，并因此在机器学习文献中近来受到关注 [26, 27, 28]。

我们证明，在对模拟和数字硬件神经网络的运行时间尺度作出现实假设的情况下，即使存在信号噪声和硬件中的器件间差异，也可以训练硬件解决 CIFAR-10 等现代数据集，速度甚至快于在图形处理器上训练软件网络。该框架的一大优势是，它只需对硬件进行最少的改动，就能让原本仅为推理而设计的硬件平台执行在线训练。

2 多路复用梯度下降

2.1 通过扰动计算梯度

我们首先作出基本假设：我们拥有一些带有可编程参数（例如权重和偏置）、能够执行推理的硬件。我们的目标是以最小改动扩展硬件，使其也能通过梯度下降进行训练。我们将展示如何配置硬件，使整个网络自动执行梯度下降，而无需反向传播。举例来说，假设我们有一套硬件

实例，用来实现图 1 所示的前馈多层神经网络。该硬件接收时变输入 $x(t)$和训练目标 $\hat{y}(t)$，具有可变参数 θ，输出推理结果 $y(t)$，并计算代价 $C(y(t), \hat{y}(t))$。为了计算该系统的梯度，我们首先在每个参数的基准值 θ_i 上添加一个小的时变扰动 $\theta_i(t)$（见图 1a 插图）。该扰动会使代价 C略微变化，而这种变化会反馈给参数。最终，这一过程将使我们能够提取系统的梯度。

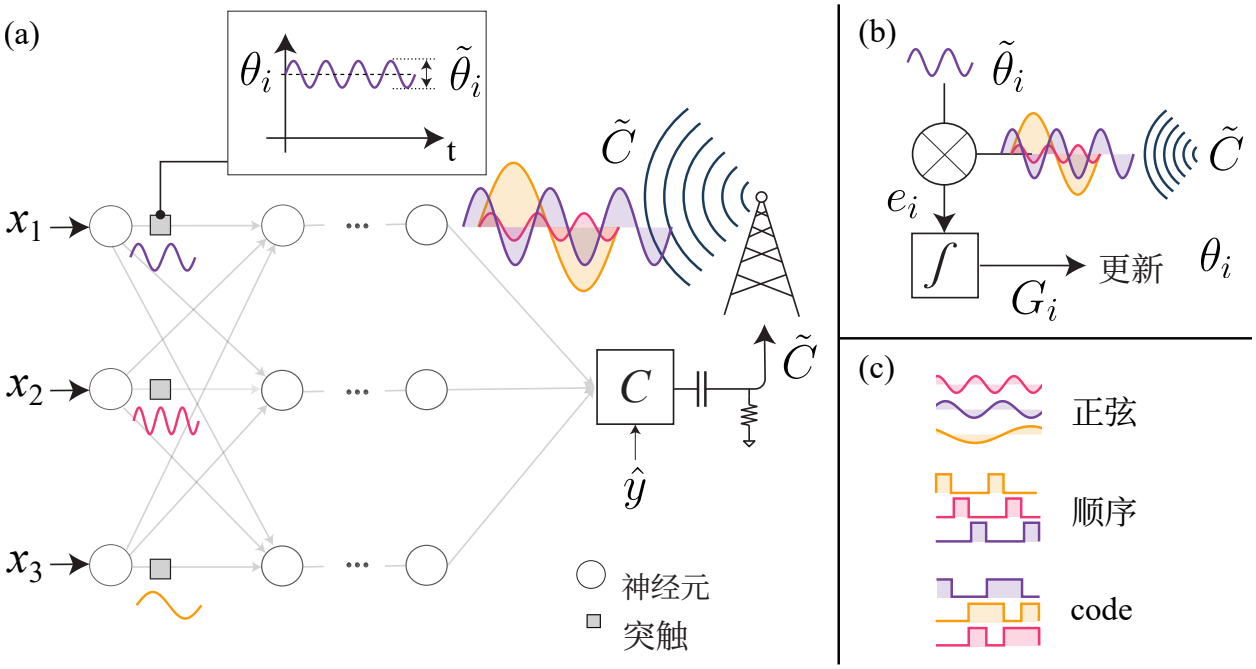


图 1：(a) 示意图展示了在前馈神经网络中，MGD框架如何利用正弦扰动示例运行。(a, 插图) 每个参数 i 都会从其基准值 θ_i 受到扰动 θ_i的轻微调制。这些扰动会导致代价 C 发生调制，并将其全局广播回所有参数。(b) 使用零差检测过程，根据 θ_i 和 C的积分乘积计算偏梯度近似值 G_i 。随后利用该偏梯度，沿梯度的近似方向更新 θ_i 。(c) 可用于此过程的扰动类型示例。

尽管扰动可以采取多种不同形式，我们首先将使用正弦扰动来描述这一过程，因为从概念上讲，这种扰动很容易理解。在这种情形下，每个参数 θ_i 都会以独特的频率 ω_i 和振幅 $\Delta\theta$进行轻微调制，从而产生扰动 $\theta_i(t) = \Delta\theta\sin(\omega_i t)$。由于每个参数都经过

调制，因此它会使 $y(t)$ 发生轻微变化，进而改变代价。因此，如果参数以频率 ω_1、ω_2、ω_3 等进行调制，那么这些相同的频率必然会~以微小调制的形式出现在代价 $C(t)$ 中，叠加在基准（未受扰动）代价值 C_0 之上，使得

$$C(t) = C_0 + \tilde{C}(t) = C_0 + \sum_i \Delta C_i \sin(\omega_i t) \tag{1}$$

~如果去掉 C_0，剩下的就是一个随时间变化的信号 $C(t) = \sum_i \Delta C_i \sin(\omega_i t)$，它仅对应参数扰动的影响。振幅 ΔC_i 就是由参数 i 的扰动 $\theta_i(t)$ 所导致的代价变化幅度。

由于相对于代价 $dC/d\theta$ 的梯度完全由偏梯度 $dC/d\theta = (\partial C/\partial\theta_1,\ \partial C/\partial\theta_2,\ .....)$ 构成，如果我们能提取每个参数对应的 ΔC_i，就能估算出完整梯度 $G = (\Delta C_1/\Delta\theta_1,\ \Delta C_2/\Delta\theta_2,\ .....)$。现在的~任务变成了从求和后的信号 $C(t)$ 中提取各个 ΔC_i。幸运的是，要提取给定的 ΔC_i，我们只需将输入扰动~ ~$\theta_i(t)$ 与 $C(t)$ 的乘积进行积分。该积分采用零差检测的形式，通过积分消除其他参数带来的无关扰动（频率）：

$$\begin{aligned} G_i &= \frac{1}{\Delta\theta_i^2}\frac{1}{T}\int_{t=0}^{T} \sum_k \Delta C_k \sin(\omega_k t)\Delta\theta_i \sin(\omega_i t)dt \\ &= \frac{\Delta C_i}{\Delta\theta_i} \quad \text{as} \quad T \to \infty \end{aligned} \tag{2}$$

其中$1/\Delta\theta_i^2$是一个归一化常数。

值 G_i 是参数 i的偏梯度近似值。G 在 $T \to \infty$ 和扰动幅度 $\Delta\theta_i$ 都趋近于零时，会趋近于精确梯度；否则，它仅为近似值。幸运的是，即使在实际的时间尺度和幅度下，G 仍包含有意义的信息，可用于执行梯度下降 [12]。

为便于说明，我们使用正弦参数扰动来描述该算法。不过，可以使用任何一组正交的零均值扰动[13], ，包括图 1c 所示的各种模拟和离散扰动。一般而言，我们将对乘积 $e_i(t) = \tilde{C}(t)\tilde{\theta}_i(t)/\Delta\theta_i^2$ 进行积分，并将其称为误差信号； G_i 则由 [1] 给出

$$G_i = \int_{t=0}^{T} \frac{\tilde{C}(t)\tilde{\theta}_i(t)}{\Delta\theta_i^2} dt \tag{3}$$

我们将在第3.4节讨论改变扰动类型的影响。还要指出的是，尽管此处描述的许多量都是时变的，但为简洁起见，在接下来的各节中，我们将省略显式的时间依赖记法。

2.2 MGD框架中的梯度下降

这里我们介绍一种在硬件中实现无模型梯度下降框架的实用方法，并将其称为复用梯度下降（MGD）。为了从硬件角度更好地理解该算法，我们将按照此前的描述，重新演示相同的计算过程，但这次从单个参数（例如硬件神经网络中的一个突触权重）的角度出发。该过程始于施加局部扰动 θ_i ，使参数的基准值 θ_i 略有变化（图1a，插图）。如前所述，这种扰动以及其他参数产生的任何其他扰动，都会使代价 C 在基准代价 C_0 的基础上发生变化，因此输出端的代价为 $C = C_0 + C$。 C 可以通过直接从 C 中减去 C_0 来提取；或者，在某些模拟场景中，也可以使用简单的高通滤波器。由此产生的 C 信号会全局广播给所有参数，因此我们的参数 θ_i 也能获取该信号。（请注意，尽管图1为便于说明展示了一座无线广播塔，但在大多数硬件平台中，这通常是通过有线连接实现的。

[1]请注意，此处以及仿真结果中， G_i 会随时间累积，且不会像方程 2 中那样除以$1/T$进行归一化。如后文所述，这样我们就可以改变积分时间，而不会对训练速率造成显著影响——换言之，可以进行长时间积分，从而沿梯度大步下降；也可以连续进行一系列较短的步进，沿梯度前进大致相同的距离。

）。不过，我们必须假设，除第 i个参数外，其他参数也会引起代价变化。对于我们的参数 θ_i而言，这些其他变化都是不需要的，必须将其滤除。如图1b所示，为了让参数 i 仅提取~自身对代价的影响，只需对其局部扰动 θ_i 与~它接收到的全局成本信号 C 的乘积进行积分。由于扰动信号具有两两正交性，这样便能分离出 θ_i所产生的贡献。根据公式3，该积分会产生偏梯度近似 $G_i \propto \Delta C_i/\Delta\theta_i$。随后，参数可以利用该 G_i 值，通过按照梯度下降步更新自身来直接降低代价

$$\theta_i \to \theta_i - \eta G_i \tag{4}$$

其中 η 是学习率。当系统的所有参数并行执行此操作时，产生的权重更新就相当于对整个网络进行梯度下降训练。由于所有参数都同时受到扰动并更新，我们将这一框架称为多路复用。

从硬件角度来看，还必须考虑一些实际问题，例如何时进行权重更新、梯度应积分多长时间，等等。为在 MGD 框架下处理这些问题，我们引入以下时间常数。

- τ_{p} 是扰动发生的时间尺度。在数字系统中，每隔τ_{p}，每个参数的扰动都会更新为新值。在模拟（连续）系统中，τ_{p} 对应于扰动的特征时间尺度。例如，若使用正弦扰动，它大致对应于这些频率所占据的总带宽。

- τ_{θ} 是梯度积分时间（即公式3中的 T ）。它决定参数更新的频率，也决定梯度近似的准确程度。

对于每个时间段 τ_θ，梯度近似 G 会被积分，并且在该时间段结束时，根据公式4更新参数。

- τ_x 控制新训练样本 $x, \hat{y}$ 呈现给硬件的频率。每经过一个 τ_x 周期，旧样本就会被丢弃并应用新样本，从而生成新的输出 y 和代价 C。

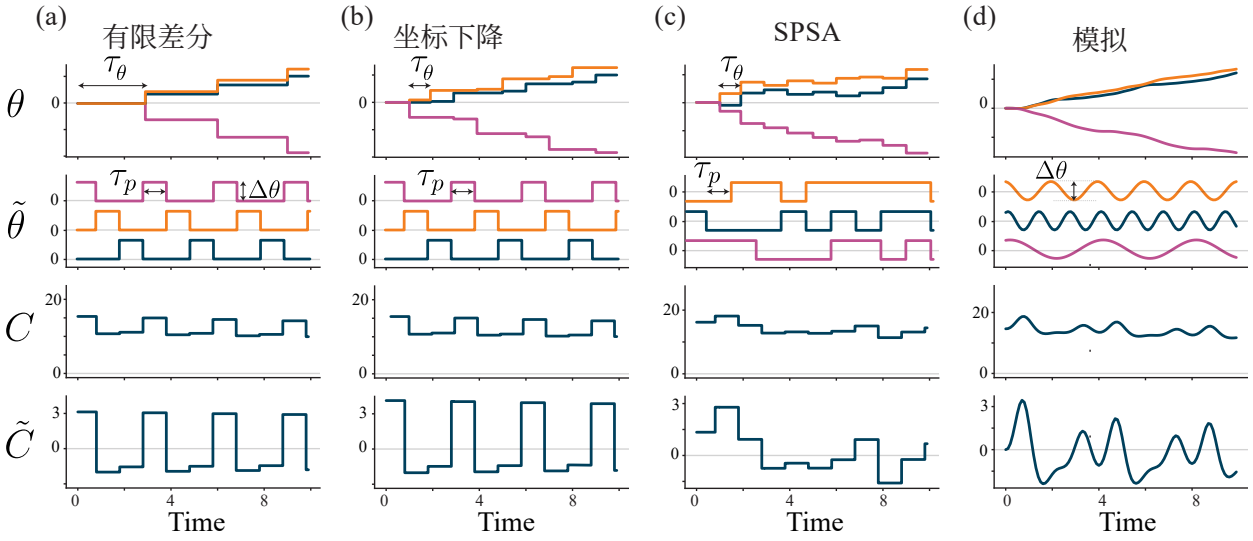


图 2：在 MGD 框架中，通过更改时间常数的取值 τ_p、τ_θ 和 τ_x，并改变扰动$\tilde{}$类型，可以实现不同的优化算法。图中展示了每隔 τ_θ更新一次的参数值 θ 、扰动 $\tilde{\theta}$及其特征时间常数 τ_p、计算得到的代价 C以及代价调制 C。改变 τ_θ和 τ_x 以及扰动类型，可使 MGD 实现多种优化算法，包括 (a) 有限差分、(b) 坐标下降、(c) 同时扰动随机近似（SPSA）和 (d) 模拟实现。上方两个图中的每种线条颜色对应本次仿真中的三个参数之一。

时间常数 τ_θ 和 τ_p 的取值会对训练的具体过程产生重大影响，而对 τ_θ 和 τ_p 的特定选择，则可在数值分析中实现某些传统算法。例如，考虑在此框架中实现前向有限差分算法。为此，我们从离散扰动开始，使得每个 τ_p 只有一个参数受到 $\Delta\theta$的扰动，并按照图 1c 所示依次扰动各参数。当参数 i 受到 $\Delta\theta_i$的扰动时，代价会变化 ΔC ，由此得到的偏梯度 $\Delta C/\Delta\theta_i \approx \partial C/\partial\theta_i$ 会存储在 G_i中。

现在，如果我们设置 $\tau_\theta = P\tau_p$，其中 P 是网络中的参数数量，那么这就完全等同于计算梯度的有限差分近似：每个 τ_p，都会近似得到梯度的一个分量；经过 $P\tau_p$后，每个偏梯度 G_i 都已测量并存储。由于 $\tau_\theta = P\tau_p$，只有在收集完 G 的所有偏梯度后，才会进行权重更新，这与有限差分算法相同。图 2a 示意了这一过程。

同样，只需修改时间常数的值，就可以简单地实现其他优化算法。例如，使用上述相同的过程，但将积分时间缩短为单个时间步，即 $\tau_\theta = \tau_p$。这与坐标下降完全对应。在这种情况下，无需存储每个 G_i 直到梯度的所有偏导数都完全汇总，而是在每个 G_i 计算完成后立即应用权重更新。从硬件角度来看，这可能比有限差分方法更具吸引力，因为可以使用 G_i 进行权重更新，随后立即将其丢弃——不同于有限差分方法，它不需要逐参数存储器来存储 G_i，直到执行权重更新。该坐标下降过程如图 2b所示。

作为第三个示例，可以仅通过改变时间常数的值和扰动形式来实现同时扰动随机逼近（SPSA） [12] 。在这种情况下， $\tau_\theta = \tau_p$，并且每隔 τ_p对每个参数施加一次随机的离散 $\{+\Delta\theta, -\Delta\theta\}$ 扰动，如图 2c所示。与坐标下降类似，这种配置无需额外的逐参数存储器，因为无需存储 G_i 值。该方法避免了对参数进行全局同步的需要——扰动无需按顺序进行，而是可以在参数处本地随机生成。

除了可以选择这些特定的优化算法之外，通过改变时间常数 τ_p 和 τ_θ，还可以实现全新的优化算法。例如，在图 2d 中，我们展示了一种在模拟系统上使用正弦扰动实现的 MGD，它不属于前面提到的任何方法。在这种

情形下， τ_{p} 对应于时间尺度 $1/\Delta f$，其中 Δf 是扰动带宽，即最大与最小扰动频率之差。此外，在模拟情形中，参数不会进行离散更新，而是由 τ_{θ} 作为积分时间常数。与离散情况不同，离散情况会先累积 G 达 τ_{θ} 时间，然后将其重置为零； θ 则会持续更新为低通滤波器的输出，该滤波器的时间常数为 τ_{θ}（参见算法 2）。

由于现代机器学习数据集由大量训练样本组成（通常有数万个）， τ_{x}（控制训练样本更换频率的时间常数）至关重要。事实上，只要恰当设置 τ_{x} ，即使硬件一次只允许输入 1 个样本，也可以实现小批量处理。批量大小由 $\tau_{\theta}/\tau_{\mathrm{x}}$ 决定，即梯度积分时间 τ_{θ} 与样本更新时间 τ_{x} 的比值。当 τ_{x} 短于 τ_{θ} 时，在单次梯度积分期间会向网络呈现多个样本。随着样本变化，梯度近似 G 会包含这些样本各自的梯度信息。这种随时间积分的过程，在算术上等同于并行累加多个样本（这通常通过图形处理器实现）。举例来说，对于一次只接受一个输入样本的硬件，如果 $\tau_{\theta} = \tau_{\mathrm{x}}$ ，批量大小为 1；但如果 $\tau_{\theta} = 4\tau_{\mathrm{x}}$，批量大小则为 4。图 3 展示了这一过程，使用的网络与图 2 相同，并采用同时离散扰动。在第 3 节中，我们将证明，批量大小高达 1000 时，MGD 仍能正常运行。

总而言之，只需设置三个时间常数和扰动类型，MGD框架便能实现多种梯度下降变体；根据所需的方法，还可以针对问题和硬件的需求定制训练。例如，正如我们在第 3.2 节所示，通过将 τ_{θ} 设置为任意长[2]，MGD可以与反向传播相匹配。与此同时，我们也证明，较短的 τ_{θ} 值同样可以取得良好效果。在实践中，写入速度限制等硬件考量可能要求采用一个

[2]当 τ_{θ} 变得任意长时，MGD 对偏导数的近似将收敛于真实的偏导数。因此，利用数字运算和反向传播实现的上述各个参数更新步骤，可以任意接近其解析定义。

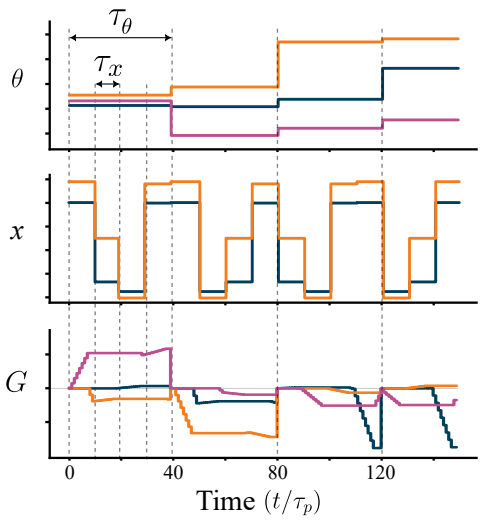


图3：示意一个包含三个参数和两个 x输入的小型网络中的批处理，该网络在包含四个样本的数据集上训练。参数 θ 每隔 τ_θ更新一次，在此期间，所有四个训练样本都会输入网络，并纳入梯度近似（批量大小为 $\tau_\theta/\tau_x = 4$）。梯度近似 G 在每个时间步累积，并在每个 τ_θ 周期结束后的权重更新过程中重置。请注意，根据公式4， θ 的更新方向与 G相反。

中间 τ_θ 值，正如我们在第4节所讨论的。这些特性使MGD框架能够采用那些在训练机器学习模型方面已取得巨大成功的相同训练技术，同时还能兼容新型和涌现式硬件。

3 MGD仿真

3.1 引言

为了评估 MGD框架的实用性，我们首先仿真了它在现代机器学习数据集上的性能。模拟器的目标并非尽可能快地在中央处理器或图形处理器上执行梯度下降，而是模拟实现 MGD 的硬件，并评估其在硬件环境中的潜在性能。具体而言，我们使用模拟器估算其速度、准确率，以及抗噪能力和对制造缺陷的耐受性。该模拟器使用 Julia 语言编写，可在中央处理器上运行

或在图形处理器上运行，并可在线获取[3]。仿真中使用的算法如算法 1 和算法 2 的方框所示，软件中的参数及其类型则列于表 1。

算法 1 离散算法

初始化参数 θ
对于 n **在**迭代次数**中执行**
    如果(n取模 $\tau_{\mathrm{x}} = 0$) **则**
        输入新的训练样本 x, $\hat{y}$
    如果(n取模 $\tau_{\mathrm{x}} = 0$) **或** (n取模 $\tau_{\theta} = 0$) **则**
        将扰动设为零 $\tilde{\theta} \leftarrow 0$
        更新基准代价 $C_0 \leftarrow C(f(x;\ \theta),\ \hat{y})$
    如果 (n模 $\tau_{\mathrm{p}} = 0$) **则**
        更新扰动 $\tilde{\theta}$
    计算输出 $y \leftarrow f(x;\ \theta + \tilde{\theta})$
    计算代价 $C \leftarrow C(y,\ \hat{y})$
    计算代价变化 $\tilde{C} \leftarrow C - C_0$
    计算瞬时误差信号 $e \leftarrow \tilde{C}\tilde{\theta}/\Delta\theta^2$
    累积梯度近似 $G \leftarrow G + e$
    如果 (n模 $\tau_{\theta} = 0$) **则**
        更新参数 $\theta \leftarrow \theta - \eta G$
        重置梯度近似 $G \leftarrow 0$

算法 2 模拟算法

初始化参数 θ
对 $t = 0$ **至** T，**步长为** dt **执行**
    如果 (t取模 $\tau_{\mathrm{x}} = 0$) **则**
        输入新的训练样本 x, $\hat{y}$
    更新扰动 $\tilde{\theta}$
    计算输出 $y \leftarrow f(x;\ \theta + \tilde{\theta})$
    计算代价 $C(t) \leftarrow C(y,\ \hat{y})$
    计算离散高通 $\tilde{C}(t) \leftarrow \frac{\tau_{\mathrm{hp}}}{\tau_{\mathrm{hp}}+dt}\left(\tilde{C}(t-dt) + C(t) - C(t-dt)\right)$
    计算瞬时误差信号 $e(t) \leftarrow \tilde{C}\tilde{\theta}dt/\Delta\theta^2$
    更新梯度近似 $G(t) \leftarrow \frac{dt}{\tau_{\theta}+dt}\left(e(t) + \frac{\tau_{\theta}}{dt}G(t-dt)\right)$
    更新参数 $\theta \leftarrow \theta - \eta G$

[3]代码可在 https://github.com/amccaugh/multiplexed-gradient-descent-paper 获取

描述	符号	模拟或数字
代价变化由扰动引起	$\tilde{C}$	both
参数扰动	$\tilde{\theta}$	both
参数	θ	both
输入样本	x	both
目标输出	$\hat{y}$	both
网络输出	y	both
Cost	C	both
未扰动基线代价	C_0	数字
梯度近似	G	both
瞬时误差信号	e	both
学习率	η	both
扰动幅度	$\Delta\theta$	both
输入样本变化时间常数	τ_{x}	both
参数更新时间常数	τ_{θ}	both
扰动时间常数	τ_{p}	数字
高通滤波器时间常数	τ_{hp}	模拟

表 1：仿真中使用的算法参数和变量

3.2 与反向传播的等价性

我们首先验证了该仿真能够针对一个示例问题最小化代价，并且在适当选择参数的情况下，它等价于通过反向传播进行的梯度下降。在这项初步比较中，我们选择通过训练一个具有 9 个参数（ 6 个权重、 3 个偏置）的 2-2-1 前馈网络来解决 2 位奇偶校验问题。我们使用非常大的 τ_{θ} 和 $\tau_{\theta} = \tau_{\mathrm{x}}$ 值对该问题进行仿真，从而在 G 中针对每个训练样本获得非常好的梯度近似。随后，我们使用为 1 的 τ_{θ}值运行相同的问题，因此每个样本的梯度近似 G 相对较差。我们测量了两项实验的训练轮次数量和时间（仿真的迭代次数），结果如图 4 所示。这里，训练轮次采用通常的定义方式——相当于网络接收数据集中的所有训练样本。

从图 4a 中按训练轮次比较这两种情形，可以观察到，取值

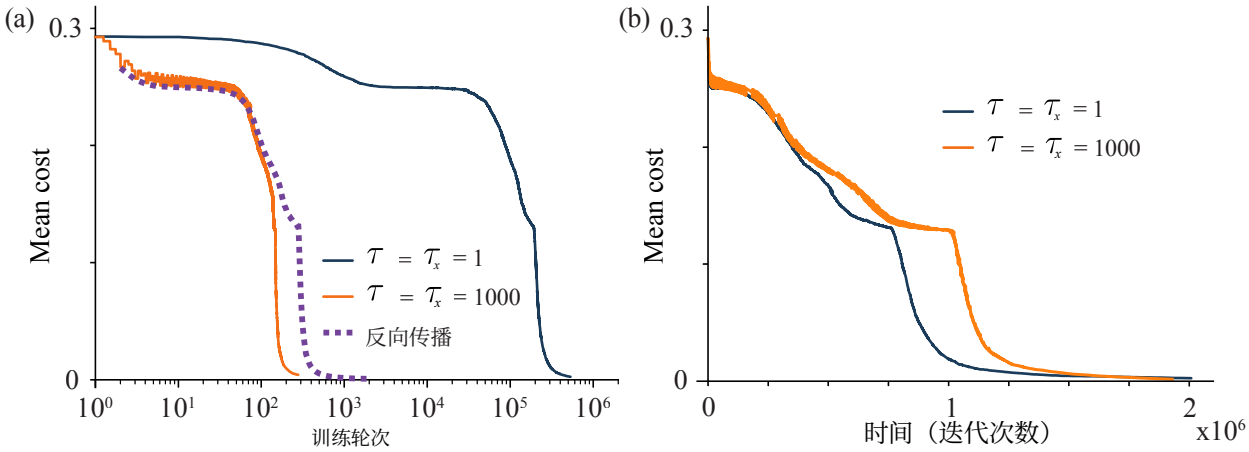


图 4：使用包含 9 个参数的 2-2-1 网络求解 2位奇偶校验（异或）问题，$\tau_p = 1$，批量大小为 $\tau_\theta/\tau_x = 1$。(a) 2-2-1 前馈网络在不同梯度积分时间 τ_θ（实线）下使用 MGD 训练，并在 1000 次随机初始化上的平均代价随训练轮次的变化。虚线表示使用反向传播训练的相同网络和数据集。(b) 与 (a) 相同的 MGD 训练数据，横轴改为时间（扰动次数 τ_p），而非训练轮次。

为 $\tau_\theta = \tau_x = 1000$ 时，系统的训练轨迹与反向传播几乎完全相同。这正如预期——对于呈现给网络的每个样本，梯度近似 G 有 1000 个时间步来积分，从而得到非常准确的估计，该估计应与通过反向传播计算出的真实梯度非常接近。然而，当 $\tau_\theta = \tau_x = 1$ 时，每个样本只有一个时间步来估计梯度，然后就要转向下一个样本。因此，为了最小化代价，这些样本必须向网络呈现更多次，导致训练轮次大幅增加。

然而，虽然 $\tau_\theta = \tau_x = 1$ 情形对样本数据的利用效率较低（需要更多训练轮次），但数据效率与运行时间之间实际上存在权衡。若绘制代价随迭代次数而非训练轮次变化的曲线，我们就能更准确地估算硬件以实际时间计训练所需的时长。如图 4b 所示，较短的 τ_θ 和 τ_x 值将代价降至最小所需的时间约为较长取值的一半。这些示例表明，尽管较长的积分时间能产生更准确的梯度近似，但也可以使用短至 τ_p 的积分时间来训练网络，这与文献 [12] 的发现一致。事实上，较短的积分时间可能

甚至在运行时间方面也可能训练得更快。

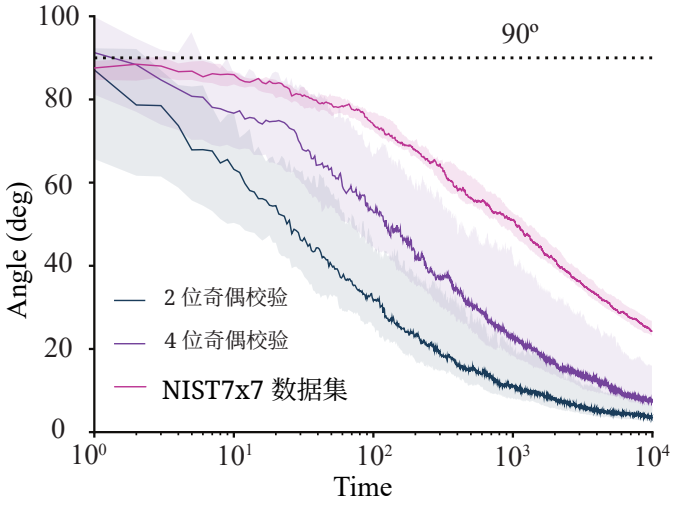


图 5：梯度近似 G 与真实梯度之间的角度随时间的变化。n 位奇偶校验网络使用了 n-n-1 网络，而 NIST7x7 网络使用了一个 49-4-4 网络。这些网络分别有 9、25 和 220 个参数，数据集分别包含 4、16 和 44136 个训练样本。实线表示 n 位奇偶校验任务的 100 次随机初始化以及 NIST7x7 数据集的 15 次随机初始化所得的角度中位数。阴影区域表示第 1 至第 3 四分位数。

为了量化较长积分时间对梯度近似准确率的影响，我们测量了梯度近似 G 随时间收敛至真实梯度 $\partial C/\partial\theta$（由反向传播计算得出）的情况。在此实验中，我们以 $\tau_\theta = \infty$ 和 $\tau_x = \tau_p = 1$ 运行仿真，使其持续对 G 进行积分，而不重置或更新参数。仿真运行期间，我们反复计算真实梯度 $\partial C/\partial\theta$ 与近似值 G 之间的角度。结果如图 5 所示，展示了 2 位奇偶校验、4 位奇偶校验和 NIST7x7 问题的解。NIST7x7 数据集是一个小型图像识别问题，旨在像素平面上识别字母 N、I、S 和 T。该数据集具有这样的特性：对于采用 Sigmoid 激活函数的 49-4-4 前馈网络，使用线性求解无法达到超过 93% 的准确率。我们选择这一网络和数据集，也是因为它们规模较小，足以进行大量不同的仿真以获取统计数据，同时问题空间足够大，使得随机解不太可能出现。

正如预期，随着时间推移，角度逐渐减小，因为 G 与真实梯度对齐。时间轴以 τ_{p} 为单位，这是该系统中最小的离散时间步。对于真实的硬件平台，这个时间步大约等于系统的推理时间。一般来说，网络的参数越多，收敛到真实梯度所需的时间就越长。

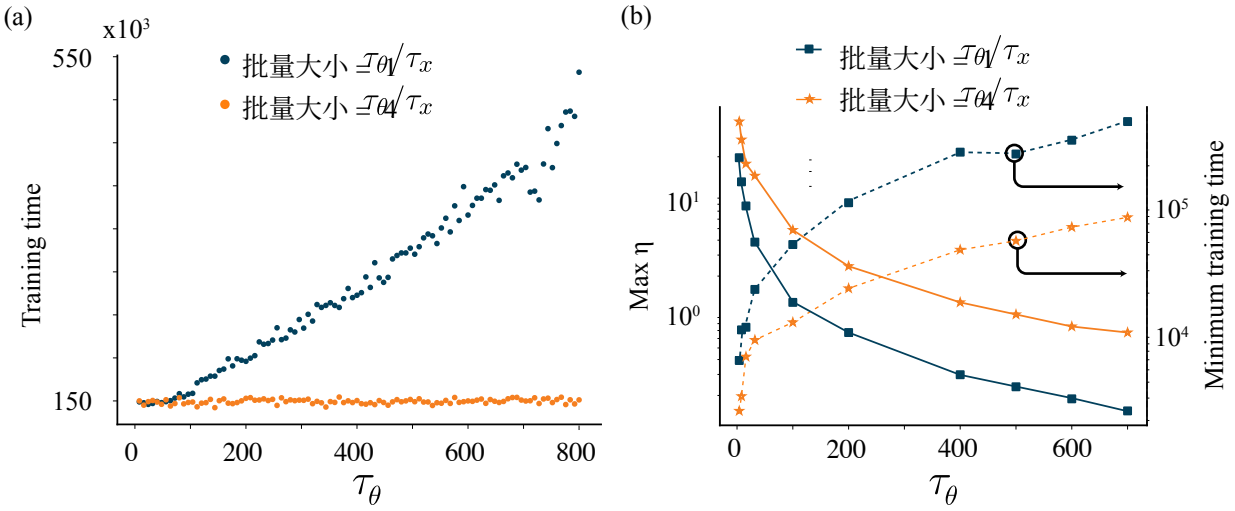


图6： τ_θ 对 2位奇偶校验（异或）问题训练时间的影响。(a) 训练时间随 τ_θ 和批量大小的变化。(b) τ_θ 对可达到的最大学习率 η 及其对应的最短训练时间的影响。

3.3 小批量处理的详细说明

接下来，我们进一步详细研究了 τ_θ 和 τ_x 如何影响训练时间。较大的 τ_θ 值会带来更准确的梯度近似，但会降低参数更新的频率。我们采用固定且较低的 η 值，训练一个 2-2-1 网络，使其解决 2 位奇偶校验（异或）问题；针对 100 组不同的随机参数初始化，我们改变 τ_θ，同时将批量大小 τ_θ/τ_x固定为 4 或 1。由于 2 位奇偶校验数据集由四组（x，$\hat{}y$）构成，$\tau_x = 4\tau_\theta$ 类似于梯度下降——在进行权重更新之前，先将全部四个样本纳入梯度近似 G。当 $\tau_x = \tau_\theta$时，网络采用批量大小为 1 的随机梯度下降（SGD）。图 6a 展示了这两种情况下训练时间随 τ_θ 变化的情况。此处，训练时间对应于总

代价 C 降至 0.04 以下的时刻，这表明问题已成功解决。当批量大小为 1 时，我们观察到，增大 τ_θ 会延长训练时间。然而，当批量大小为 4 时，增大 τ_θ 对训练时间几乎没有影响。

与任何训练过程一样，在较高的 η 值下，训练可能会变得不稳定，无法完成任务。由于图 6a 中的结果仅针对固定学习率，我们还希望考察 τ_θ 对可实现的最大 η的影响。在这里，我们将“最大 η”定义为：网络在 100 次随机初始化中至少有 50 次成功解决 2 位奇偶校验问题时所对应的最大学习率。如图 6b 所示，随着 τ_θ增大，最大 η 会降低，从而导致最短训练时间变长。无论批量大小较大还是较小，结果都是如此，不过较大的批量大小总体上可实现更高的 η 值。

从这些结果中，我们推断，相较于针对单个样本收集准确的梯度，针对所有训练样本得到较差的梯度近似更有用。换句话说，与其等待很长时间以获得极其准确的梯度，然后迈出一大步，不如连续迈出一系列较小（但不那么准确）的步子。这对硬件而言是一个重要结论，因为它表明，在 MGD 中实现有效的梯度下降过程，并不一定需要额外的内存来存储准确的高位深度梯度值。请注意，在我们的实现中，G 会随时间累积，因此公式4中的参数更新大小 ηG会随积分时间成比例增长。这意味着，当 τ_θ 较大时，沿梯度方向的有效步长也较大；因此，在固定 η 的情况下，训练速率仍大致保持不变。如果情况并非如此，那么每当 τ_θ 翻倍时，我们也需要将 η 减半，才能维持大致相同的训练速率。

3.4 模拟与数字扰动

参数扰动可以采取许多不同的形式，只要其振幅较低，并且时间平均值两两正交，或者在统计意义上不相关——

即可 [13]。我们实现了四种扰动：正弦扰动、顺序离散扰动、离散码扰动和随机码扰动。正弦扰动与图 1a 所示的类似，其中每个参数都分配了一个独特的频率。顺序离散扰动是指按顺序逐个扰动参数的情况，如第 2.2 节有限差分实现中所述，每次扰动 $+\Delta\theta$。对于每个参数，“码”扰动是每隔 τ_{p} 个时间步，同时对 $\{-\Delta\theta, +\Delta\theta\}$ 进行离散扰动。我们称之为码扰动，是因为它们类似于无线通信技术中使用的码分复用（扩频）技术。码扰动有两种形式：第一种由一组预定义的两两正交方波函数构成，其取值为 $\{-\Delta\theta, +\Delta\theta\}$。每种扰动模式都是一个确定性序列，并且任意两个参数都不会使用相同的序列。例如，现代蜂窝电话通信中使用的沃尔什码就属于此类。第二种由随机生成的 $\{-\Delta\theta, +\Delta\theta\}$ 序列构成，这些序列两两不相关。我们称其为“统计正交”。统计正交的情形比确定性正交˜码的效率稍低，因为多个参数的扰动会在 C 中相互干扰得更多——任何有限长度的扰动样本都会存在非零相关性，但该相关性会随样本量增加而趋近于零。不过，采用统计正交形式可以在本地随机生成扰动。在硬件实现中，这些扰动可能非常有用，因为它们具有扩频特性，而且外部来源的单频噪声不太可能干扰反馈信号。

为了比较不同扰动类型之间的训练性能，我们将四种不同的扰动类型应用于上一节所述的同一个 2-2-1网络，以解决 2位奇偶校验问题。具体而言，我们旨在展示训练既可以以纯模拟方式进行，也可以以纯数字方式进行。在实践中，许多系统具有模拟数字混合特性，因此可以采用组合方法。

图 7 展示了使用不同扰动类型解决 2位奇偶校验问题时的训练时间分布。

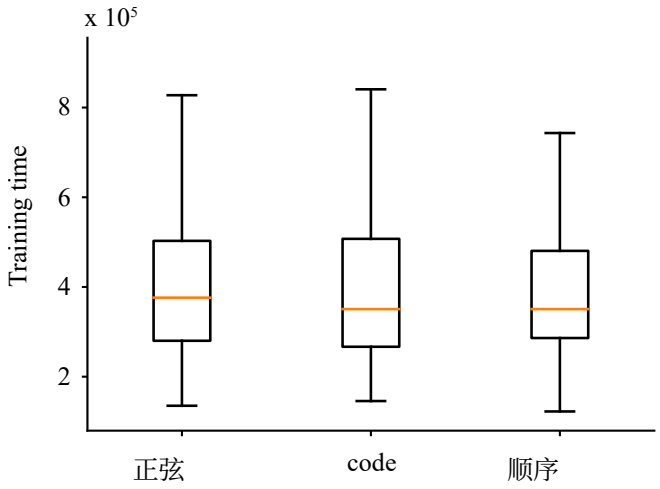


图 7：通过测量在 2-2-1网络上训练 2位奇偶校验（异或）问题所需的时间，展示各种扰动类型之间的等效性。每个箱线图代表 100 次随机初始化的结果。超参数为 $\tau_{\mathrm{x}} = 250$, $\tau_{\theta} = 1$, $\eta = 0.05$, $\tau_{\mathrm{p}} = 1$ （离散）， $\Delta f = 0.3$ （模拟）

.

正弦扰动的带宽设为 $1/2\tau_{\mathrm{p}}$。正如预期，我们发现不同扰动类型在训练速度方面大致相当。考虑到来自 C 的˜反馈具有有限带宽，且所有参数都必须共享这一带宽，这种等效性就说得通了——无论采用何种编码（扰动）方案，反馈中传递给参数的信息都会受到这一有限带宽的限制。

3.5 在有噪声或不完美硬件上的操作

新兴硬件平台中存在的制造缺陷和信号噪声给当前的硬件训练技术带来了重大挑战。例如，权重参数可能偏离设计值，或以非确定性方式更新，导致模型与实际硬件之间出现差异，从而对训练产生不利影响 [29, 30, 9]。例如，文献 [30], 中的模拟网络在 MNIST 数据集上达到了 97.6% 的准确率，但将预训练模型直接迁移到硬件上的衍射光学网络实现，在未进行某些原位训练的情况下，经过

直接迁移后，其准确率仅为 63.9%；而文献 [31] 表明，3% 的器件变化可能导致分类准确率下降 7%，且未采用误差校正。文献 [9] 展示了模型参数中的微小差异如何在一般的层次化问题中被放大。在其简化问题中，模型参数 0.5% 的误差经过 20 层后会导致输出产生 30% 的误差。对于离线训练而言，解决这些非理想因素可能十分繁琐。例如，在训练期间可能必须定期测量权重的实际值 [29], ，并仔细测量这些非理想因素，将其纳入模型 [32, 31], ；或者，可以使用低位深度对权重进行量化，并选择适当的位深度，使系统即使存在器件间差异，仍能被准确建模和控制。这在忆阻器交叉阵列 [33] 或相变材料 [34], 等硬件平台中很常见，在这些平台上可以实现 6–8 位的位深度。

在这里，我们研究了三种可能影响硬件系统的非理想因素和噪声：(1) 输出代价上的随机噪声 C_{noise}，(2) 参数更新上的随机噪声 θ_{noise}，(3) 激活函数中的逐神经元缺陷，即每个神经元都具有一个随机缩放并偏移的S形激活函数，且该函数不随时间变化。除非另有说明，这些测试均使用前文所述的 49-4-4 网络（含 220 个参数）在 NIST7x7 数据集上进行，并采用 $\tau_{\mathrm{x}} = \tau_{\theta} = 1$ 。

在第一项测试中，我们向代价添加了均值为零、标准差为 σ_{C} 的高斯噪声，并在每个时间步施加该噪声，使得噪声 $C(t) = C_{\mathrm{ideal}}(t) + C_{\mathrm{noise}}(t;\, \sigma_{\mathrm{C}})$。例如，在光学硬件中，这可能是由激光波动引起的噪声。代价中的噪声也等同于任何其他影响累积梯度 G 的高斯零均值噪声。图 8a 展示了随着 σ_{C} 增大，不同学习率对训练时间的影响。对于给定的学习率，噪声低于某一阈值时，训练时间几乎不变。然而，随着代价噪声增大，训练时间最终会延长，并且训练最终会停止收敛。

为了了解这种噪声会如何影响优化学习的最短训练时间

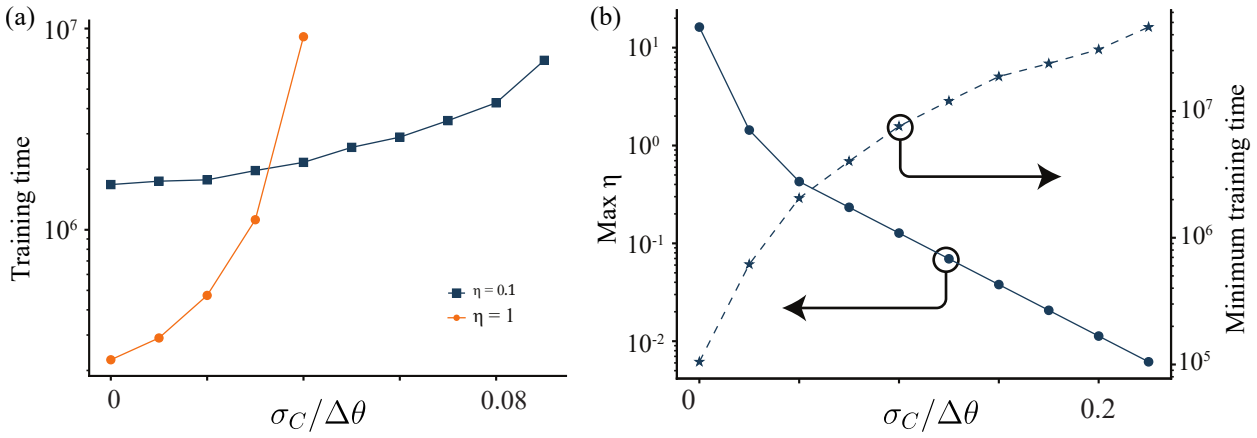


图 8：对代价信号施加噪声的影响。(a) 训练时间与 σ_C 的关系，其中 σ_C 是添加到代价信号中的高斯噪声幅度 C（相对于扰动幅度$|\theta|$进行归一化）。训练时间是 10 次随机参数初始化达到 $> 80\%$ 准确率所需时间的中位数。(b) 噪声对最大 η 和训练时间的影响。最大 η 是使 10 次随机初始化中有 $> 80\%$ 收敛的最高学习率；右侧纵轴显示该最高学习率下对应的训练时间。

率，我们还测量了在一系列 σ_C 下可达到的最大 η 值。图 8b 显示了最大 η 值随代价噪声的变化，以及相应的最短训练时间。趋势表明，代价噪声 σ_C越低，我们就能将 η 调得越高，训练速度也越快。反过来说，只需降低学习率，就能补偿系统中的大量代价噪声。

在下一项测试中，我们分析了含噪参数更新对训练过程的影响。在这项实验中，每当更新任何参数时，更新都会包含一个随机施加的偏差。因此，更新规则变为

$$\theta \leftarrow \theta - \eta G + \theta_{\text{noise}} \tag{5}$$

其中 θ_{noise} 服从均值为零、标准差为 σ_θ的高斯分布，并由 $\Delta\theta$归一化，使得 $\theta_{\text{noise}} \sim \mathcal{N}(0,\ \sigma_\theta/\Delta\theta)$。这种噪声常见于没有闭环反馈的模拟存储器中 [35, 36]。

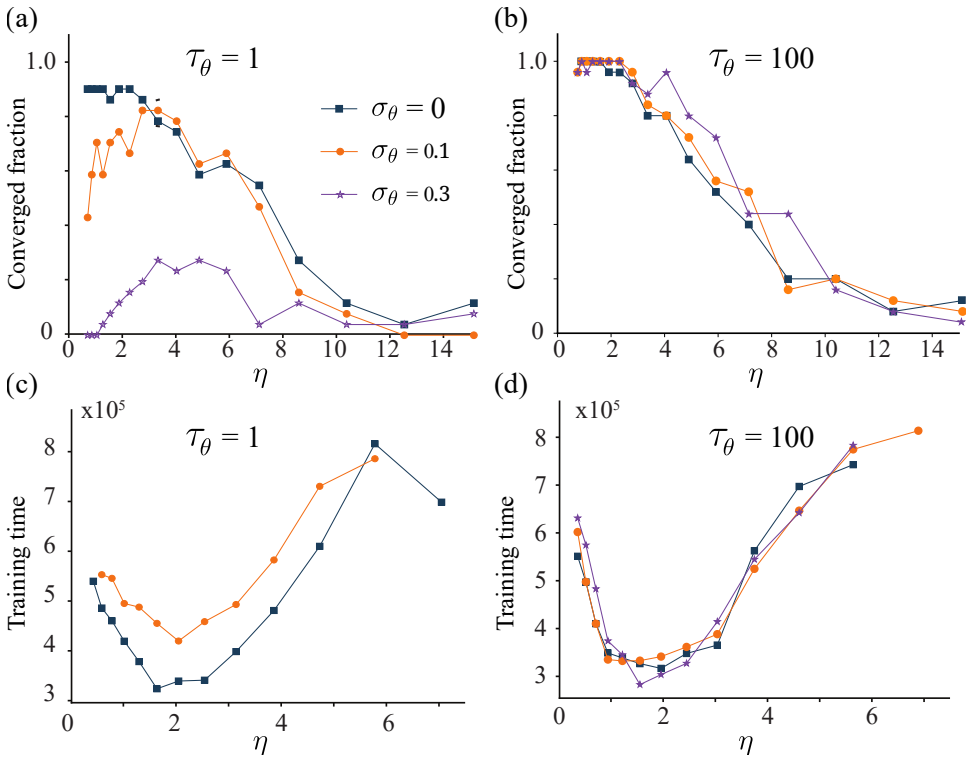


图 9：在 2-2-1 前馈网络中求解异或时，不同噪声幅度 σ_θ下，带噪声（随机）参数更新的影响。(a、b) 参数更新噪声对训练收敛概率的影响，该概率随学习率 η变化而变化。与 (b) $\tau_\theta = 100$ 相比，噪声更有可能阻止 (a) $\tau_\theta = 1$ 收敛。收敛定义为在 5×10^7 次迭代内达到 93% 的准确率。统计数据来自 25 次随机参数初始化。(c、d) 参数更新噪声对训练时间的影响，该时间随学习率 η 以及积分时间 (c) $\tau_\theta = 1$ 和 (d) $\tau_\theta = 100$变化。仅显示具有 $> 50\%$ 收敛率的 η值组——在 (c) 中，$\sigma_\theta = 0.3$ 没有满足该标准的 η 值。

我们发现，较大的 σ_θ 值会完全阻止收敛（图 9a）。有趣的是，在存在这种噪声时，增大 η 实际上可以改善问题的收敛情况，如图 9a 中的 $\sigma_\theta = 0.1$ 和 $\sigma_\theta = 0.3$ 曲线所示。我们认为，这可能是因为当 η 值非常小时，θ_{noise} 会压过公式 5 中极小的 ηG 。增大η 项，可以避免 ηG 被 θ_{noise}淹没。显然，当 η 值非常大时，通常的梯度下降不稳定性便开始占据主导地位，收敛趋近于零。对于给定的 η，我们发现，较小的 σ_θ 值会略微增加训练时间，但其影响不如直接改变学习率 η显著（图 9c）。

降低 θ_{noise} 影响的另一种方法是增加梯度的积分时间。当 τ_θ 增大时， G 的值会累积更长时间，从而按比例增大。例如，当 τ_θ 增大100倍时， ηG 的值也会增大100倍，这意味着式5中 θ_{noise} 的相对影响缩小到原来的1/100。结果如图9b、d所示，即使 σ_θ 值最大，对结果的影响也很小。

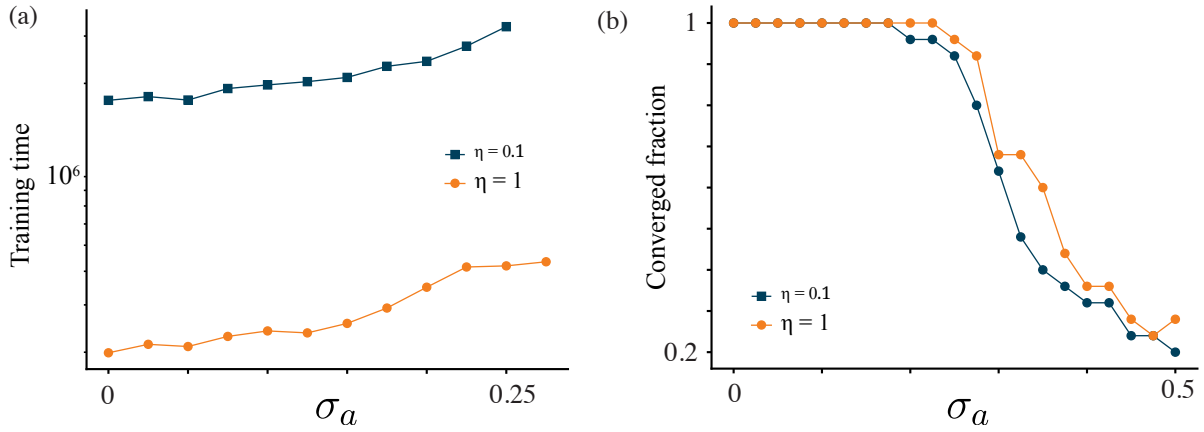


图10：为每个神经元的sigmoid激活函数添加随机偏移和缩放的影响。(a) 训练时间与 σ_a（激活函数偏移量和缩放的标准差）的关系。(b) 收敛比例与逻辑函数参数标准差的关系。结果为25种不同随机网络初始化和激活函数随机化的统计数据。收敛定义为在仿真时间内达到80%的准确率。

在最后一次测试中，我们分析了在神经元激活函数中引入“缺陷”的影响。此时，神经元激活函数不再是完全相同的 sigmoid 函数，而是具有固定的随机偏移和缩放，且这些变化在时间上保持静态。这些变化旨在模拟硬件中可能存在的器件间差异，例如模拟超大规模集成电路神经元中的差异 [37]。每个神经元的 sigmoid激活函数 k 都被修改为一般逻辑函数 $f_k(a) = \alpha_k(1 - e^{-\beta_k(a-a_k)})^{-1} + b_k$。所有变化都服从高斯分布；缩放因子 α_k, β_k 的标准差为 σ_a ，均值为 1；偏移因子 a_k, b_k 的标准差也为 σ_a ，但均值为零。

如图 10a 所示，向网络的激活函数中添加缺陷，对训练时间的影响相对较小。即使激活

设置		参数					准确率				
Task	网络	$\|\theta\|$	τ_θ	τ_p	η	批量大小	10^4 步	10^5 步	10^6 步	10^7 步	反向传播
2 位奇偶校验	2-2-1	9	1	1	5	1	100%	100%	100%	100%	100%
N-I-S-T	49-4-4	220	1	1	3	1	38%	81%	94%	97.7 %	99.8%
N-I-S-T	49-4-4	220	1	1	0.5	1	22%	45%	93%	98.7%	99.8%
Fashion-MNIST	两层卷积神经网络	14378	1	1	9	1000	34.2%	66.3%	79.3%	83.5%	88.6%
Fashion-MNIST	两层卷积神经网络	14378	10	1	9	1000	34.3%	66.3%	79.2 %	83.4%	88.6%
Fashion-MNIST	两层卷积神经网络	14378	100	1	9	1000	35.3%	66.3%	77.7 %	84.7%	88.6%
Fashion-MNIST	两层卷积神经网络	14378	1000	1	9	1000	35.3%	59.6%	79.1 %	86.1%	88.6%
CIFAR-10	3层卷积神经网络	26154	1	1	9	1000	12%	23%	43.8%	60.7%	68%

表2：MGD在四种不同数据集上的训练表现。文中展示了设置、MGD参数以及达到的测试准确率。最后一列列出了同一网络通过反向传播训练所达到的最佳准确率，以供比较。

函数发生了较大变化（$\sigma_a = 0.25$），网络完成 NIST7x7 数据集的训练所需的时间也仅约为原来的两倍。然而，偏差过大（$\sigma_a > 0.25$）会导致训练难以收敛。这可能是因为输出神经元的缩放和偏移程度过大，已无法产生正确的输出。

总体而言，我们发现，对于真实硬件系统中可能出现的合理程度的不同类型噪声和制造缺陷，MGD都具有鲁棒性。总体来说，这些非理想因素会影响网络的训练速度，但不会妨碍其解决预期问题。

3.6 现代数据集结果

为了评估 MGD 对更大规模机器学习问题的可扩展性和适用性，我们针对不同的网络架构和超参数，在多种任务上比较了 MGD 与反向传播。表2比较了使用 MGD 和反向传播时，在不同数据集及各种超参数设置（τ_θ、 τ_p、 η、批量大小）下得到的准确率，其中τ_x 固定为 1。为了进行有效比较，反向传播实验采用了不带动量的基础随机梯度下降（SGD）优化器。两种策略均使用均方误差（MSE）代价函数。这样设置可以避免更复杂策略带来的混杂影响，不过值得注意的是，MGD 能够实现其中一些较复杂的特性（例如动量、丢弃法等）。所有实验中，两组结果使用的网络架构都相同。所用的网络架构

并非为达到最先进的准确率而选择，而是选用了规模较小的网络，以便快速测试和进行统计分析。若要进一步提高准确率，则需要采用更先进的网络架构（例如增加层数）或优化器（例如加入动量）。为了进行有意义的比较，我们测量了 MGD 过程在固定时间步数后所达到的准确率。这使我们能够估算第4节中不同硬件配置下的实际训练时间。

CIFAR-10 网络由 3 个卷积层和最大池化层组成，后接一个全连接层。卷积滤波器为 3×3，输出通道数分别为 16、32 和 64，步幅为 1，输出采用 ReLU 激活函数。每个卷积层后都接一个 2×2最大池化层。最后的全连接层将卷积层的 256 个输出转换为 10 个类别，且未使用 Softmax。Fashion-MNIST 网络结构类似，由两个卷积层和最大池化层组成，后接一个 (32×10) 全连接层，且未使用 Softmax。用于解决 NIST7x7 和异或问题的网络是采用 Sigmoid 激活函数的全连接前馈网络。为尽可能公平地比较表2中的结果，我们使用所列批量大小和多种 η值训练网络，以最大化每一行的反向传播准确率。报告的反向传播准确率，是在测试的所有 η 值中获得的最高中位数准确率；该中位数取自五次随机初始化，每次训练 2500 个轮次（远在准确率收敛之后）。

通过在 2 位奇偶校验、NIST7x7、Fashion-MNIST 和 CIFAR-10 数据集上训练神经网络，我们发现 MGD 达到的解的准确率接近反向传播。然而，即使经过 10^7 步，我们仍观察到，MGD 训练的网络准确率略低于反向传播所能达到的水平。这可能与梯度近似未满足收敛判据有关，原因是 η 值固定不变——例如，SPSA 过程背后的理论仅保证在学习率渐近趋近于零时收敛 [12]。总体而言，我们观察到，较高的学习率能在训练早期取得更高的准确率，而较低的学习率则能在

更长时间内取得更高的准确率（例如，参见 NIST7x7 结果）。因此，定制学习率可能有助于实现更优的训练时间和准确率。

仿真还表明，对于此处所示较大数据集，在固定的 η 值下，改变 τ_θ 对 MGD 的最大准确率影响甚微，而增大 τ_θ 会使最终准确率略有提升。这可能是因为较长的 τ_θ 值能够产生更准确的梯度估计，并使网络在系统接近代价函数的局部极小值时更好地优化权重，不过还需要更多实验和统计数据来验证这一效应是否普遍存在。幸运的是，我们的结果表明，这一效应相对较小，因此硬件设计者可以灵活决定执行权重更新的频率。

这些结果明确表明，MGD 是一种可行的技术，可用于在新兴硬件平台上训练真实的机器学习数据集。以往的理论研究主要集中于收敛性证明以及不同精度下的梯度近似计算 [12, 13] ，或集中于简化问题，例如线性激活函数 [38]。尽管这些理论结果显然十分重要且不可或缺，但它们不足以深入揭示这些优化技术应用于实验硬件平台上的真实机器学习问题时的适用性。与此同时，以往的实验和仿真研究大多集中于小规模问题；对于这类问题，达到 100% 的准确率轻而易举，例如异或 [14, 15], 、少位奇偶校验 [16, 20, 21], 、鸢尾花数据集 [24]或其他类似的玩具问题 [17, 18, 25]。这些问题并不能代表较大的机器学习数据集；后者通常包含数万个训练样本，并在拥有更多参数的网络上进行训练。

这些结果超越了以往的任何一项研究，阐明了这些技术对于更大规模的现代机器学习数据集的实用性，并实现了一个框架，可通过特定的硬件参数连接不同的优化技术（ SPSA、有限差分、近似梯度下降）。

还值得注意的是，随机权重变化（RWC）算法已在忆阻器网络中实

现过 [23, 39] ，尽管表面上与MGD相似，但其功能截然不同。RWC并非一种近似梯度下降技术，因为权重更新并不按代价变化的幅度进行缩放，而是根据随机扰动能否降低代价来决定保留还是舍弃。因此，与我们在MGD框架中实现的优化技术相比，该算法在参数数量增加时的扩展性较差。

4 硬件考量

4.1 时间常数

	HW1	**HW2**	**HW3**	**反向传播**
τ_x **（输入样本更新时间）**	100 纳秒	1 ns	10 皮秒	n/a
τ_p **(扰动时间)**	1 ms	10 ns	200 ps	n/a
τ_θ **(参数更新时间)**	1 ms	1 µs	200 ps	n/a
2位奇偶校验训练时间（10^4 **步**）	20 s	200 微秒	4 µs	70 毫秒†
Fashion-MNIST 训练时间（10^6 **步**）	33 分钟	20 毫秒	400 微秒	54 秒*
CIFAR-10 训练时间（10^7 **步**）	5.6 小时	200 毫秒	4 ms	480 秒*
硬件示例	芯片在环， 集成光子学 采用热光调谐[40, 11]	存算一体器件 [41]， 模拟VLSI [42]	超导器件 [43]， 无热谐振 硅光调制器 [44]	†中央处理器/*图形处理器

表3：使用 MGD 并基于估算的硬件时间常数计算出的近似训练时间。数据是基于与表2相同的数据集和网络架构估算的。最后一列显示了使用标准GPU或中央处理器通过反向传播达到相同准确率所需的时间。

尽管 MGD框架具有通用性，但不同硬件平台在实际应用中可能有不同的考量因素，这些因素决定了各种时间常数的最佳选择。这里我们讨论硬件实现中可能遇到的问题，并指出可能存在的权衡。

扰动速度（τ_p）： 在许多硬件平台中，直接扰动参数可能很困难，或者由于存储器耐久性或更新速度方面的顾虑而不理想。不过，没有必要直接扰动参数；可以改为扰动一个与该参数串联的独立元件。例如，在光子马赫-曾德尔干涉仪阵列中，权重可以通过响应较慢的（~毫秒级）热相移器或机械相移器实现，而微小扰动则可以通过响应较快的（~纳秒级）电光调制器实现，

例如耗尽型 [44]或光机 [45] 调制器。或者，在忆阻器系统中，可以通过一个与忆阻器串联的亚阈值晶体管来实现扰动。

要使 MGD 等扰动式技术正常运行， τ_{p} 应慢于系统的推理时间 [13] ，其中包括成本计算和广播过程。如果 τ_{p} 过短，就可能导致输入扰动与由此产生的成本反馈不同步。

参数更新（τ_{θ}）；在硬件上执行梯度下降时，必须多次更新参数。对于长度为 T的训练周期，权重将更新 T/τ_{θ} 次。对于某些硬件平台，减少参数更新次数可能更有利，例如参数寿命有限、更新时间较长，或更新过程需要消耗大量能量时。幸运的是，表2显示，对于参数数量较多的神经网络，可以增大 τ_{θ} ，而不会显著影响整体训练速度或最终准确率。

更改训练样本（$\tau_{\mathbf{x}}$）： τ_{x} 的取值受训练样本输入网络的速度限制。对于大多数技术而言， τ_{θ} 或 τ_{x} 不会成为限制因素，因为传统计算机或 FPGA 能够以纳秒时间尺度提供 x 和 $\hat{}y$ 个样本。不过需要注意的是，如果 τ_{x} 短于硬件的推理时间，MGD 将无法正常运行；这很可能是 τ_{x}实际面临的限制。 τ_{x} 也可用于控制批量大小（由 $\tau_{\theta}/\tau_{\mathrm{x}}$给出）。任务和数据集的具体情况可能会决定所需的批量大小，而这可能会对 τ_{x}产生一定影响，例如 τ_{θ} 的最小允许值非常长时。

成本计算： 代价计算仅在系统输出端执行，因此用于计算代价的硬件可以成本更高、结构更复杂。在集成电子系统中，代价的实现可以很简单，但它占用的芯片面积可能比其他元件相对更大。不过，对于非常特殊的硬件平台，这种成本计算仍可能

难以在片上执行。可以通过采用非标准代价函数，或使用计算机在芯片外计算代价来解决这一问题。不过，芯片外计算的输入和输出可能会限制成本计算的速度，也会限制˜ C的全局广播速度，并可能成为某些硬件系统中的另一个速度瓶颈。

根据文献，表3列出了一些可在硬件中实现的合理时间常数估计值，以及根据表2结果求解基准任务所需的相应时间。最后一行列出了文献中的一些硬件平台示例，这些平台可实现这组参数。将这些结果与表3最后一列进行比较后可以看出，采用对新兴硬件的现实估计，MGD 的速度可能显著快于当前使用反向传播的实现。

4.2 模拟和数字实现

MGD框架的模拟硬件实现与数字硬件实现之间有几个显著差异。离散情况要求在网络输出˜处设置一个存储元件，用于存储 C_0 （采样保持），并通过简单的减法运算计算 $C = C - C_0$。网络还需要安排一些时间步来测量 C_0。这意味着，对于 $\tau_\theta = \tau_p$这一简单情况，训练整个硬件系统只需额外设置一个存储元件，位置在网络代价输出处。

然而，在 $\tau_\theta > \tau_p$ 的情况下，每个参数都需要额外的存储元件（例如模拟电容器或离散存储器）来执行梯度积分。模拟情形要求在每个参数元件处设置一个低通滤波器，并在网络输出端设置一个单独的˜高通滤波器，以将 C 转换为 C。这些滤波器可以用简单的模拟电路（如 RC 或 LR滤波器）实现。不过，我们注意到，在某些情况下，使用连续高通滤波器可能会导致实现问题。例如，如果 η 过大， θ 的快速变化可能产生不需要的频率分量，这些分量会与扰动输入混合，并可能对梯度近似产生负面影响。此外，如果数据集是离散的，跳变

˜在 x 中的跳变可能会通过 C 和 C 传播高频噪声，再次对梯度近似产生负面影响。仿真结果表明，原则上，MGD 可用于具有模拟或数字特性的不同硬件平台，以进行网络训练。

5 讨论

MGD训练框架与无线通信系统之间可以进行一个有趣的类比。在无线通信中，手机用户必须能够从包含多个用户信号的接收广播中提取与自身通信相关的信号。MGD的运行方式与此类似——每个参数都类似于一个独立的˜手机用户，而全局代价广播 C 则类似于蜂窝基站发射机。˜广播信号 C 对所有参数均可用，每个参数从该广播信号中提取与自身相关的（梯度）信息——这类似于语音数据。

无线通信中使用的编码和复用技术通常统称为通信中的“多址接入”技术，其中包括频分复用、时分复用和码分复用等多种形式。类似地，在MGD中，这些复用技术分别对应于第3.4节所讨论的不同扰动类型。例如，正弦扰动可以视为一种频分复用：每个“用户”（参数）都有一个包含与其相关信息的独特频率，并且必须执行时间积分，以便从包含许多其他无关信号的 C信号中提取该信息。同样，如第3.4节所述，离散 $\{-\Delta\theta, +\Delta\theta\}$ 扰动类型类似于现代手机中使用的码分复用技术。

需要注意的是，对于固定带宽的广播，所有复用技术都具有相同的名义频谱效率。然而，实际系统的具体情况会极大地影响各种复用技术的有效性。例如，由于码分复用会在较宽的时间窗口和扩频范围内对用户的信号进行编码，因此它

已被证明相比时分复用对多径失真和定时误差具有更强的鲁棒性。

同样，前文所述的各种扰动类型，可能会因硬件网络和环境的具体情况而表现得更好或更差。新开发的通信技术也可能直接适用于真实硬件系统中的MGD训练。在硬件中使用该方法的关键在于，每个参数都可以像手机一样运行——作为独立设备，仅凭本地可用信息和全局广播接收器，处理器和内存。这与大多数其他训练算法不同；在那些算法中，误差信号必须传播至整个网络 [46]。在软件中，这种上游信息流微不足道，但这意味着硬件神经元必须：(1) 配备额外内存以处理误差信号；(2) 分别执行前向推理和反向误差传播所需的不同操作；(3) 以同步方式运行，才能正确传播误差。

在脉动阵列中，通常不鼓励采用“广播”，因为通信的时间和能量与输入数量成正比，系数为 CV^2 （其中 C 随输入数量增加）。此外，电容越大，通信所需的放大器也越大，因此还需要消耗更多能量和占用更多面积。然而，在其他类型的模拟硬件或涌现式硬件中，广播可能是一种更节能、更省空间的过程。例如，在光学硬件中，全局光信号可以通过简单的平板波导模式传输。或者，在模拟电气硬件中，可以使用单个电气平面˜将 C 信号传回各个参数。

MGD框架在生物学上也可能比其他算法更加合理。目前有许多算法正在开发中，旨在解开大脑如何学习之谜，并以生物学为灵感，寻找更适合硬件的方案[46, 47, 48]。从实用性和生物学合理性的角度来看，反向传播的一大问题是，前向（推理）阶段和反向（学习）阶段需要采用不同类型的运算。MGD则不然：在MGD中，训练

是在与推理相同的配置下进行的。从生物学类比的角度来看，全局代价˜信号 C 可以建模为大脑中的某种准全局化学信号。

未来，MGD算法可扩展为使代价不再对每个突触全局适用，而是针对特定的神经元/突触簇。大脑中存在具有这类特性的化学信号。扰动所需的时间尺度可能与生物学上观察到的效应相吻合，例如短时程可塑性（分钟） [49, 50] 和突触噪声 [51]。MGD也与“三因子”学习规则一致，越来越多的实验证据支持这类规则 [50]。此外，尽管本工作未对此进行探讨，MGD应当也可在脉冲模型中发挥作用。已有研究在小型脉冲网络中探索了类似算法，它们利用神经元电导的扰动 [52], 、概率性突触传递 [53] 或脉冲时序到达的随机性质 [54] 进行基于奖励的学习。

6 结论

本文表明，在新兴硬件具有现实时间尺度的情况下，与在标准GPU/CPU上采用反向传播相比，借助MGD进行训练在达到解的实际运行时间方面可快几个数量级。MGD框架允许使用单一的全局代价信号和局部参数更新来实现多种优化算法。所使用的算法类型（例如有限差分、坐标下降、SPSA等）可通过调节MGD时间常数进行调整，甚至可根据需要在训练过程中调整。由于它是一种无模型扰动技术（有时称为零阶优化），因此适用于广泛的系统——它既可应用于模拟和数字硬件平台，也可在存在噪声和器件缺陷的情况下使用。这克服了使用基于新兴技术的硬件平台时面临的一项主要障碍，因为这类平台通常难以训练。

展望未来 ，在大型和小型硬件系统上探索 MGD 应用的机会很多。

下一步最直接的做法是在现有硬件上测试训练性能，例如在光子 [40]或忆阻器交叉阵列硬件上采用芯片在环过程。即使不重新设计硬件，也可以进行这项测试，因为 MGD 在芯片在环配置下只要求能够输入样本、捕获推理输出并修改参数。采用这种方式时，速度很可能受系统输入输出限制。例如，可以通过外部计算机直接向硬件注入扰动，再由同一台计算机捕获代价的变化、执行梯度近似并计算权重更新。这样就能在不改动硬件的情况下测试该算法。最终，为了克服输入输出限制，可以设计并实现局部电路，以进行自主在线训练。虽然本文未对此进行详细研究，但在这种情况下，所考察的许多硬件平台在能耗方面也可能提升几个数量级。

在调整这项技术及其超参数方面，也有很大的改进空间。本文几乎没有进行超参数优化或正则化，因此通过考察丢弃法、动量和正则化等技术，可能还有进一步优化的机会。此外，值得关注的是在大型数据集上考察该算法的节点扰动版本 [55]，这可能显著减少扰动元素的数量，并加快训练过程。

虽然本文仅考察了前馈网络，但已有研究表明，扰动技术可用于小规模训练循环神经网络 [56]、脉冲网络 [53] 以及其他非标准网络；未来仍需证明这些技术在当前受关注的问题中的实用性。这将适用于各种神经形态硬件及其他物理神经网络。

归根结底，MGD 非常适合直接通过本地自主电路在片上实现。尽管在无需外部计算机参与、真正实现自主学习之前，仍有大量工作要做，但这样的设备将真正

为远程和边缘计算应用带来新的可能，使其能够利用实时数据进行现场训练。

7 致谢

作者感谢美国国家标准与技术研究院（NIST）NCSG 社区提供的宝贵讨论。本研究由美国国家标准与技术研究院（https://ror.org/05xpvk416）和科罗拉多大学博尔德分校（https://ror.org/02ttsq026）资助。

参考文献

[1] G Hinton Y LeCun、Y Bengio。深度学习。《自然》，521(7553):436–444，2015年5月。

[2] Catherine D. Schuman、Thomas E. Potok、Robert M. Patton、J. Douglas Birdwell、Mark E. Dean、Garrett S. Rose 和 James S. Plank。神经形态计算与硬件神经网络综述。2017年5月。

[3] Wilfried Haensch、Tayfun Gokmen 和 Ruchir Puri。下一代深度学习硬件：模拟计算。《*IEEE*会刊》，107(1):108–122，2019年1月。

[4] Paul A Merolla、John V Arthur、Rodrigo Alvarez-icaza、AndrewS Cassidy、Jun Sawada、Filipp Akopyan、Bryan L Jackson、Nabil Imam、Chen Guo、Yutaka Nakamura、Bernard Brezzo、Ivan Vo、Steven K Esser、Rathinakumar Appuswamy、Brian Taba、Arnon Amir、Myron D Flickner、William P Risk、Rajit Manohar 和 Dharmendra S Modha。《一款具有可扩展通信网络和接口的百万级脉冲神经元集成电路》。《科学》，345:668，2014。

[5] Mike Davies、Narayan Srinivasa、Tsung-Han Lin、Gautham Chinya、Yongqiang Cao、HarshaChoday、Georgios Dimou、Prasad Joshi、Nabil Imam、Shweta Jain、Yuyun Liao,

Chit-Kwan Lin、Andrew Lines、Ruokun Liu、Deepak Mathaikutty、Steve Mccoy、Arnab Paul、Jonathan Tse、Guruguhanathan Venkataramanan、Yi-Hsin Weng、Andreas Wild、Yoonseok Yang 和 Hong Wang。Loihi：一种具备片上学习功能的神经形态多核处理器；Loihi：一种具备片上学习功能的神经形态多核处理器。*IEEEMicro*，38，2018。

[6] Karlheinz Meier。《混合信号通用神经形态计算系统》。载于*2015*年*IEEE*国际电子器件会议（*IEDM*），2015年。

[7] A. Mehonic 和 A. J. Kenyon。类脑计算需要总体规划。《style id='2'>《自然》，604：255–260，2022年4月。

[8] Neil C Thompson、Kristjan Greenewald、Keeheon Lee 和 Gabriel F Manso。深度学习的计算极限。arXiv:2007.05558，2020年。

[9]Logan G. Wright、Tatsuhiro Onodera、Martin M. Stein、Tianyu Wang、Darren T. Schachter、Zoey Hu 和 Peter L. McMahon。通过反向传播训练的深度物理神经网络。《style id='2'>《自然》，601：549–555，2022年。

[10] J. Kiefer 和 J. Wolfowitz。回归函数最大值的随机估计。《style id='2'>《数学统计年刊》，23(3)：462–466，1952年。

[11]Yichen Shen、Nicholas C. Harris、Scott Skirlo、Mihika Prabhu、Tom Baehr-Jones、Michael Hochberg、Xin Sun、Shijie Zhao、Hugo Larochelle、Dirk Englund 和 Marin Soljacic。利用相干纳米光子电路进行深度学习。《自然·光子学》，11：441–446，2017 年。

[12] James C. Spall。使用同步扰动梯度近似的多变量随机逼近。*IEEE*《自动控制汇刊》，37：332–341，1992。

[13] Amir Dembo 和 Thomas Kailath。无模型分布式学习。*IEEE*《神经网络汇刊》，1(1)：58–70，1990。

[14] T. Matsumoto 和 M. Koga。用于模拟神经网络的新型学习方法。*ElL*，26(15):1136，1990年9月。

[15] Marwan Jabri 和 Barry Flower。权重扰动：用于模拟VLSI前馈与循环多层网络的最优架构与学习技术。*IEEE* 神经网络汇刊，3(1):154–157，1992年。

[16] J. Alspector、R. Meir、B. Yuhas、A. Jayakumar 和 D. Lippe。用于模拟VLSI神经网络学习的并行梯度下降方法——第5届神经信息处理系统国际会议论文集。载于*NIPS' 92*：第*5*届神经信息处理系统国际会议论文集，第836–844页，1992年。

[17] David B. Kirk 和 Douglas Kerns。多维梯度下降的模拟VLSI实现。载于 《神经信息处理系统进展》*5*（*NIPS 1992*），第789–796页，1992年。

[18] Yutaka Maeda、Hiroaki Hirano 和 Yakichi Kanata。基于同时扰动的神经网络学习规则及其硬件实现。神经网络，8(2):251–259，1995年1月。

[19] Gert Cauwenberghs。一种学习连续时间轨迹的模拟VLSI循环神经网络。*IEEE* 神经网络汇刊，7(2):346–361，1996年。

[20] Hiroshi Miyao、Kazuhiro Noguchi、Masafumi Koga 和 Takao Matsumoto。模拟神经网络的多频振荡学习方法：其在学习型大规模集成电路中的实现。日本电子与通信（第三部分：基础电子科学），80(5)，1997。

[21] Antonio J. Montalvo、Ronald S. Gyurcsik 和 John J. Paulos。面向具有片上学习功能的通用模拟VLSI神经网络。*IEEE* 神经网络汇刊，8(2)：413–423，1997。

[22] Yutaka Maeda 和 Toshiki Tada。采用同时扰动实现具备学习能力的脉冲密度神经网络的 FPGA 实现。*IEEE* 神经网络汇刊，14(3):688–695，2003年5月。

[23] Shyam Prasad Adhikari、Hyongsuk Kim、Ram Kaji Budhathoki、Changju Yang 和 Leon O. Chua。采用忆阻器桥突触的多层神经网络基于电路的学习架构。*IEEE* 电路与系统汇刊*I*：常规论文，62(1):215–223，2015年1月。

[24] ChunhuaWang、Lin Xiong、JingruSun 和Wei Yao。采用权重同时扰动训练的忆阻器神经网络。非线性动力学，95(4):2893–2906，2019年。

[25] Saumil Bandyopadhyay、Alexander Sludds、Stefan Krastanov、Ryan Hamerly、Nicholas Harris、Darius Bunandar、Matthew Streshinsky、Michael Hochberg 和 Dirk Englund。具备加速训练功能的单芯片光子深度神经网络。第1–21页，2022年8月。

[26] AtılımG¨unes Baydin、Barak A.Pearlmutter、DonSyme、FrankWood 和 PhilipTorr。《无需反向传播的梯度》，2022。

[27] David Silver、Anirudh Goyal、Ivo Danihelka、Matteo Hessel 和 Hado van Hasselt。《通过方向梯度下降进行学习》。载于国际学习表征会议（*ICLR*）论文集，2022。

[28] Mengye Ren、Simon Kornblith、Renjie Liao 和 Geoffrey Hinton。使用局部损失缩放前向梯度，2022。

[29] Brian Hoskins、Mitchell Fream、Matthew Daniels、Jonathan Goodwill、Advait Madhavan、Jabez Mcclelland、Osama Yousuf、Gina Adam、Wen Ma、Tung Hoang、Mark Branstad、Muqing Liu、Rasmus Madsen、Martin Lueker-Boden 和 Martin Lueker。《电阻式神经网络原型验证系统》。*2021*年国际神经形态系统会议，5。

[30] Tiankuang Zhou、Xing Lin、Jiamin Wu、Yitong Chen、Hao Xie、Yipeng Li、Jingtao Fan、Huaqiang Wu、Lu Fang 和 Qionghai Dai。采用可重构衍射处理单元进行大规模神经形态光电计算。《自然·光子学》，15：367–373，2021。

[31] Saumil Bandyopadhyay、Ryan Hamerly、Ryan Hamerly 和 Dirk Englund。面向可编程光子学的硬件纠错。《光学》，第 *8* 卷，第 *10* 期，第 *1247-1255* 页，8：1247–1255，2021。

[32] Amir Khosrowshahi、Casimir Wierzynski、MichaelR.DeWeese、MichaelY.-S.Fang 和 Sasikanth Manipatruni。考虑元件误差的光学神经网络设计。《光学快报》，第 *27* 卷，第 *10* 期，第 *14009-14029* 页，27：14009–14029，2019。

[33] Miao Hu、Catherine E.Graves、CanLi、YunningLi、NingGe、EricMontgomery、Noraica Davila、Hao Jiang、R. Stanley Williams、J. Joshua Yang、Qiangfei Xia 和 John Paul Strachan。利用点积引擎进行基于忆阻器的模拟计算和神经网络分类。《先进材料》，30:1705914，2018 年。

[34] Stefano Ambrogio、Pritish Narayanan、Hsinyu Tsai、Robert M. Shelby、Irem Boybat、Carmelo Di Nolfo、Severin Sidler、Massimo Giordano、Martina Bodini、Nathan C.P. Farinha、Benjamin Killeen、Christina Cheng、Yassine Jaoudi 和 Geoffrey W. Burr。使用模拟存储器进行等精度加速神经网络训练。自然，558：60–67，2018。

[35] Stefano Ambrogio、Simone Balatti、Antonio Cubeta、Alessandro Calderoni、Nirmal Ra- maswamy 和 Daniele Ielmini。HfOx 电阻式开关存储器中的统计波动：第一部分——置位/复位变异性。*IEEE*电子器件汇刊，61：2912–2919，2014。

[36] Jubin Hazra、Maximilian Liehr、Karsten Beckmann、Sarah Rafiq 和 Nathaniel Cady。改善65纳米CMOS集成的基于HfO2的纳米级RRAM器件的存储窗口/电阻可变性权衡。第2019年10月卷。电气与电子工程师学会公司，2019年10月。

[37] Cory Merkel 和 Dhireesha Kudithipudi。神经忆阻系统的片外训练方法比较。第2015年2月卷，第99–104页。IEEE计算机学会，2015年2月。

[38] Justin Werfel、Xiaohui Xie 和 H. Sebastian Seung。线性前馈网络中随机梯度下降的学习曲线。《神经计算》，17(12):2699–2718，2005年12月。

[39] Changju Yang、Hyongsuk Kim、Shyam Prasad Adhikari 和 Leon O. Chua。一种采用反向传播与随机权重变化算法进行混合学习的电路神经网络。《传感器》*2017*年，第*17*卷，第*16*页，17(1):16，2016年12月。

[40] Alexander N. Tait、Thomas Ferreira De Lima、Ellen Zhou、Allie X. Wu、Mitchell A. Nahmias、Bhavin J. Shastri 和 Paul R. Prucnal。《神经形态光子网络 using 硅光子权重库》。《科学报告》，7，2017。

[41] Geoffrey W Burr、Robert MShelby、Abu Sebastian、Sangbum Kim、Seyoung Kim、Sev- erin Sidler、Kumar Virwani、Masatoshi Ishii、Pritish Narayanan、Alessandro Fumarola、Lucas L Sanches、Irem Boybat、Manuel Le Gallo、Kibong Moon、Jiyoo Woo、Hyunsang Hwang 和 Yusuf Leblebici。使用非易失性存储器进行神经形态计算。2:89–124，2017。

[42] Y. Kohda、Y. Li、K. Hosokawa、S. Kim、R. Khaddam-Aljameh、Z. Ren、P. Solomon、T. Gokmen、S. Rajalingam、C. Baks、W.Haensch 和 E. Leobandung。无需辅助的真正模拟神经网络训练芯片。技术文摘——国际电子器件会议，*IEDM*，2020年12月：36.2.1–36.2.4，2020年12月。

[43] Michael Schneider、Emily Toomey、Graham Rowlands、Jeff Shainline、Paul Tschirhart 和 Ken Segall。Supermind：超导电子学在神经形态计算中的潜力综述。《超导科学与技术》，35:053001，2022。

[44] Erman Timurdogan、Cheryl M. Sorace-Agaskar、Jie Sun、Ehsan Shah Hosseini、Alek- sandrBiberman 和 MichaelR.Watts。超低功耗热稳定硅调制器。《自然·通讯》，5:1–11，2014。

[45]Mark Dong、Genevieve Clark、Andrew J Leenheer、Matthew Zimmermann、Daniel Dominguez、Adrian J Menssen、David Heim、Gerald Gilbert、Dirk Englund 和 Matt Eichenfield。采用兼容低温、覆盖可见光至近红外的200毫米CMOS架构实现高速可编程光子电路。《自然·光子学》，16:59–65，2022。

[46] Timothy P. Lillicrap、Adam Santoro、Luke Marris、Colin J. Akerman 和 Geoffrey Hinton。《反向传播与大脑》。《自然综述：神经科学》*2020*年*21*卷第*6*期，21(6):335–346，2020年4月。

[47] Friedemann Zenke 和 Emre O. Neftci。神经形态基底上的脑启发学习。《*IEEE*会刊》，109:935–950，2021年5月。

[48] Benjamin Scellier 和 Yoshua Bengio。平衡传播：弥合基于能量的模型与反向传播之间的差距。《计算神经科学前沿》，11:24，2017年5月。

[49] Andrea Soltoggio. 短时程可塑性作为远端奖励学习中的因果假设检验。《生物控制论》，109:75–94，22015。

[50] 行为时间尺度上的资格迹与可塑性：新赫布三因子学习规则的实验支持。《神经回路前沿》，0:53，2018年7月。

[51] Benjamin A. Rowland、Anthony S. Maida 和 Istvan S. N. Berkeley。突触噪声作为实现权重扰动学习的一种手段。《18》(1):69–79，2007年3月。

[52] Ila R. Fiete 和 H. Sebastian Seung。通过对电导施加动态扰动，在脉冲神经网络中进行梯度学习。《物理评论快报》，97:048104，2006年7月。

[53] H. Sebastian Seung。通过强化随机突触传递，在脉冲神经网络中进行学习。《神经元》，40(6):1063–1073，2003年12月。

[54] Xiaohui Xie 和 H. Sebastian Seung。通过强化不规则脉冲发放，在神经网络中进行学习。《物理评论*E*：统计物理、等离子体、流体及相关跨学科主题》，69(4):10，2004年。

[55] Barry Flower 和 Marwan Jabri。权重求和神经元扰动。载于 *NIPS'92*：《神经信息处理系统》第五届国际会议论文集，第 212–219 页，1992 年。

[56] Gert Cauwenberghs。用于监督学习和优化的快速随机误差下降算法。载于 《神经信息处理系统进展》第 *5* 卷（*NIPS 1992*），第 244–251 页，1992 年。